Conflict-Free Container Routing in Mesh Yard Layouts

Jianyang Zeng[*] Wen-Jing Hsu[*][†]

Abstract

Container terminals play an important role in global cargo transportation and they have become an essential intermodal interface between the sea and the land. In the container terminal, the service area is often arranged into rectangular blocks, which leads to a mesh-like path topology. We present a mathematical model for general container routing in mesh yard layouts. Based on this model, a simple container routing algorithm guaranteeing freedom of conflicts is then presented. The algorithm works by carefully choosing suitable containers' speeds such that the containers using the same junction will arrive at different points in time, and hence incur no conflicts; meanwhile, high routing performance can be achieved. The task completion time and the requirements on timing control during the container routing are also presented. Numerical results verify that our routing scheme has good performance and is free of conflicts.

Key words: Container, Routing Algorithm, Mesh Yard Layout, Conflict-Free, Time Requirement, Number Theory.

[*]Nanyang Technological University, Nanyang Avenue, Singapore 639798. Email: zengjy@gmail.com, hsu@ntu.edu.sg.

[†]Corresponding author.

1 Introduction

Container terminals play an important role in global cargo transportation and they have become an essential intermodal interface between the sea and the land. Nowadays the growth rate of container trade worldwide has reached 9.5%, and it is also expected that about 90% of all liner freight will be shipped in containers (Liu *et al.* 2002). Generally, the core operations in a container terminal can be grouped into the following three processes (Chen 2003).

(a) **Quay area operations:** Inbound containers are unloaded from the coming vessels and outbound containers are loaded to the vessels through quay cranes (QC).

(b) **Transfer operations:** Containers are delivered from the storage yard to the berth side by container carriers, such as prime movers, yard trucks, or AGVs (Automated Guided Vehicles).

(c) **Yard area operations:** The storage locations of containers are planned and sequenced, or containers are shuffled and redistributed over different storage blocks.

Due to the limited available resources, such as yard cranes, container carrier in yards, etc., yard area operations have been the bottleneck of port operations. At the same time, shuffling or redistributing all containers among different storage blocks is also important, since in order to improve the handling efficiency of the whole container terminal system, we need to sort containers according to their classification (Kim *et al.* 2000, Chen 2003). In this paper, we will only focus on the container routing issue of the yard operation.

Four design concepts of *automated container terminals* (ACT) have been introduced to meet the growing demand of marine transportation (Liu *et al.* 2002): *automated container terminal using automated guided vehicles* (AGV-ACT), a *linear motor conveyance system* (LMCS-ACT), an *overhead grid rail system* (GR-ACT), and a *high-rise automated storage and retrieval system* (AS/RS-ACT).

In AGV-ACT, AGVs are used to transfer containers in the yard instead of using manually operated trucks or other carriers. AGVs have been used successfully in the Delta Port terminal at Rotterdam. Part of this terminal is shown in figure 1. The Port of Hamburg and Thamesport of England are also planning to implement an Automated Guided Vehicle System (AGVS) in their new container terminal premises (Liu *et al.* 2002, Ioannou *et al.* 2000, Chan 2000).

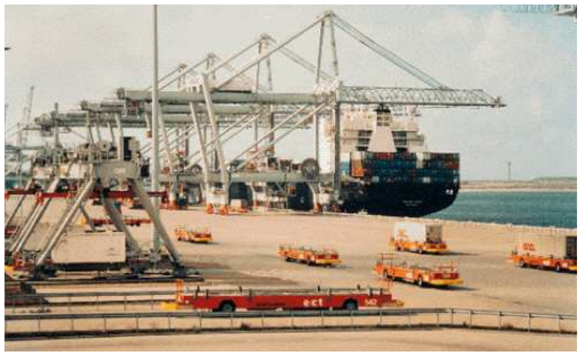

Figure 1: Automated Guided Vehicles in the Delta Port Terminal at Rotterdam [IOANNOU *et al.* 2000].

Besides being used in small scale manufacturing systems, sorting systems or assembly plants, linear motors can also be used in port and terminal operation. The linear motor conveyance system (LMCS) based ACT (LMCS-ACT) is the technologies that use the automated shuttles driven by linear motors to transfer containers in the terminal. A conceptual model of container yard using LMCS is shown in figure 2. A platform is put above the stator through a rotor, and is driven by an array of linear motor underneath it. A structure of LMCS has been constructed and tested in Eurokai Container Terminal (Liu *et al.* 2002, Ioannou *et al.* 2000).

In order to utilize the yard space more efficiently, Sea-Land and August-Design have designed a grid rail (GR) system (GR-ACT) for delivering the containers in the yard (Liu *et al.* 2002, Ioannou *et al.* 2000). As shown in figure 3, containers in GR-ACT are caught and transferred by shuttles which move along an overhead monorail in the whole terminal. The shuttles are driven by the linear induction motors above them.

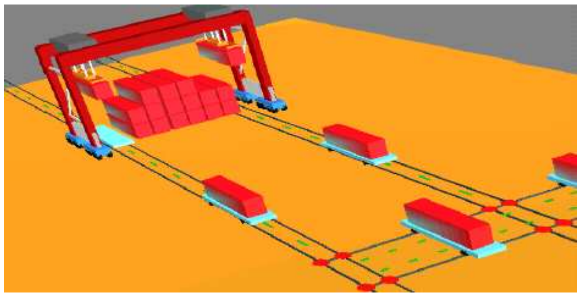

Figure 2: A conceptual prototype of LMCS [IOANNOU *et al.* 2000].

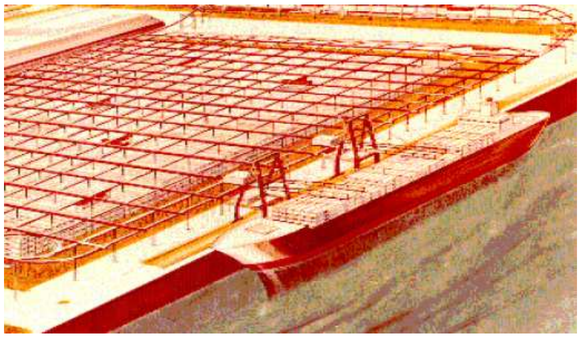

Figure 3: An example of GR-ACT [IOANNOU *et al.* 2000].

The AS/RS based ACT system can automatically deposit and retrieval containers from the storage locations. With high space utilization and productivity, the AS/RS system has become increasingly popular in the past decade. Generally, an AS/RS system consists of four components: storage cells, vertical platforms, horizontal platforms, and I/O stations (Chen *et al.* 2003), as shown in figure 4.

Due to the shapes and stacks of containers, each unit of the storage area in the yard is often arranged into a rectangular block, which leads to a mesh-like path topology (Qiu and Hsu 2000, Qiu *et al.* 2002, Zeng and Hsu 2003).

In all the four modes of ACTs, no matter whether the carrier of containers is AGV, shuttle, or platform, because they move through the junctions, the conflict issue must be addressed. In order to examine the conflict problem of the container routing in a mesh-like yard, we propose the following model for AGV, LMCS, GR, and AS/RS based ACTs, as shown in figure 5.

Figure 4: Structure of automated storage/retrieval system (AS/RS) in container yards.

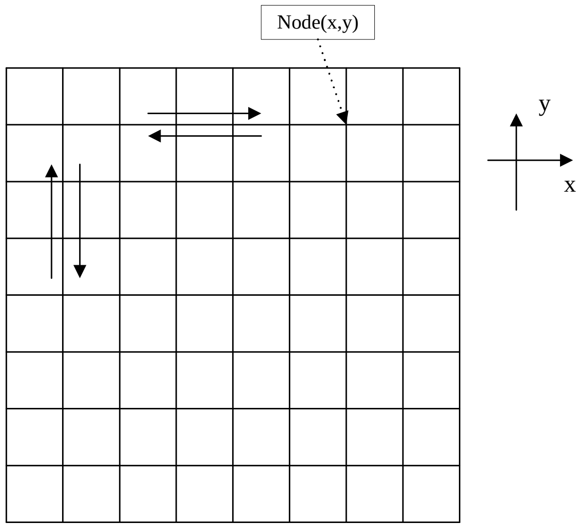

Figure 5: Abstract model of container routing in the mesh yard.

In this model, each unit square represents a storage block. There are in total $(N-1) \times (N-1)$ blocks, namely $N-1$ blocks in each column and $N-1$ blocks in each row. So there are totally $N \times N$ junctions in the mesh model. Each line denotes a bidirectional path way between adjacent blocks, whose width in one direction only admits one container to pass. Generally, each block has one Pick up-Drop off station (or *P/D station* for short). In real-world mesh yard, the P/D station is located at a special position on the boundary of the block. For convenience of analysis, we assume the P/D stations are also situated at the junction positions in our abstract model of container routing in the mesh yard. In Section 7, we will show that the model can be easily extended to the real-world case as long as the regularity of mesh layout is guaranteed.

According to the above setting, a junction and the associated P/D stations are collectively regarded as a node. Each node is assigned coordinates (x, y) as its *address* or *ID*, where x and y represent respectively the row and column IDs. This mesh layout is modeled by a graph $G = (V, E)$. The $N \times N$ vertices of the graph represent junction nodes, and the bi-directional edges represent two paths between two adjacent junction nodes, and the length of each edge is a constant. In this abstract model, the carriers of containers, such as AGVs, shuttles, or platforms are omitted, and each container is regarded as a point moving along the path in this mesh topology.

Although there are some important details for container routing, such as the size of the junction, the radius of 90° turn, and the length of the container, etc. (Huang and Hsu 1994, Qiu and Hsu 2001, Qiu *et al.* 2002), the abstract model will simplify the general mesh yard and focus only on the issue of the conflicts in routing. In Section 7, we will discuss how to extend this abstract model to incorporate realistic considerations.

In this paper, we divide the time into discrete units. Based on such a time division and the above abstract model of terminal yards, a simple algorithm for container routing is presented and timing control requirement is analyzed. The key idea lies in making use of the regularity of the mesh, and hence the regularity of points of time when containers arrive at the intersections. By choosing a suitable speed for the container along different directions, we can ensure that no conflicts among the containers will occur. We also analyze the algorithm in terms of the task completion time and requirements on the routing precision controls. By our design, the container can advance directly to their destinations, unlike in the paper by Qiu and Hsu (2000), where containers firstly are routed to intermediate places in order to reach their final destinations, and hence high performance of our scheme can also be ensured.

The remainder of the paper is organized as follows. In Section 2 some preliminaries are given. Section 3 gives the routing algorithm and the time control criteria to avoid conflicts. In Section 4, we analyze the performance of the routing algorithm. Section 5 gives an explicit method to derive the timing controls. Numerical experiments are conducted in Section 6. Finally, Section 7 discusses possibilities of relaxing certain constraints and suggests directions for future study.

2 Preliminaries

We assume that the time can be divided into discrete units of time, and that each container always reaches every junction node at some discrete point on time. It is reasonable for us to make this assumption because the distance between two adjacent nodes is a constant, and we can adjust the speed of the containers to let them arrive at the junctions at multiples of the unit time.

We also make the following assumptions for the abstract container routing model of mesh yard layout. In Section 7, we will show how to relax these assumptions in the practical applications.

(a) Each container on the mesh layout can be regarded as one point.

(b) When a container arrives at its destination, it enters a buffer area and leaves the mesh grid.

(c) All blocks in the mesh yard have equal sizes, and the P/D station is located at the same position as the junction.

Based on the abstract container routing model of mesh yard layout, we formally define the following.

***Definition* 1 (Job for the Mesh Path Layout).** A job for the mesh path layout is identified by an ordered pair $((PX, PY), (DX, DY))$, where (PX, PY) represents the address of the pickup station, (DX, DY) represents the address of the drop-off station, and $(PX, PY) \neq (DX, DY)$.

Assume that each job has a distinct origin and also a distinct (but different) destination, and in each job only one container is handled.

***Definition* 2 (Job Set for the Mesh Path Layout).** For the mesh path layout, a job set M denoting a set of k jobs, where $2 \le k \le \lfloor \frac{N^2}{2} \rfloor$, is defined as follows:
$M = \Big\{ ((PX_i, PY_i),\, (DX_i, DY_i)) \; |1 \le PX_i,\, PY_i,\, DX_i,\, DY_i \le N, \text{ for } i = 1, 2, ..., k \Big\}.$

Since we assume that each job only handles one container, the number of jobs k equals to the number of containers in the given job set.

According to the positions of the origins and destinations of jobs, any given job set M can be divided into three subsets, denoted by M_x, M_y, M_{xy} respectively, such that

$M_x = \Big\{ ((PX_i,\, PY_i),\, (DX_i,\, DY_i)) \; |DX_i \ne PX_i,\, DY_i = PY_i, \text{ for } i = 1, 2, ..., k \Big\}.$

$M_y = \Big\{ ((PX_i, PY_i),\, (DX_i, DY_i)) \; |DY_i \ne PY_i,\, DX_i = PX_i, \text{ for } i = 1, 2, ..., k \Big\}.$

$M_{xy} = \Big\{ ((PX_i, PY_i),\, (DX_i, DY_i)) \; |DY_i \ne PY_i,\, DX_i \ne PX_i, \text{ for } i = 1, 2, ..., k \Big\}.$

We also divide M_{xy} into two subsets, denoted by M_{xy+} and M_{xy-}, such that

$M_{xy+} = \Big\{ ((PX_i, PY_i),\, (DX_i, DY_i)) \; |DY_i > PY_i,\, DX_i \ne PX_i, \text{ for } i = 1, 2, ..., k \Big\}.$

$M_{xy-} = \Big\{ ((PX_i, PY_i),\, (DX_i, DY_i)) \; |DY_i < PY_i,\, DX_i \ne PX_i, \text{ for } i = 1, 2, ..., k \Big\}.$

Accordingly, we have the following notations:

A_x : the set of containers that carry out jobs in M_x;

A_y : the set of containers that carry out jobs in M_y;

A_{xy} : the set of containers that carry out jobs in M_{xy};

A_{xy+} : the set of containers that carry out jobs in M_{xy+};

A_{xy-} : the set of containers that carry out jobs in M_{xy-};

***Definition* 3 (Direction of Containers).** $\overrightarrow{v}$ is a unit vector which represents the direction of a given container, where $\overrightarrow{v} \in \{+\overrightarrow{x}, -\overrightarrow{x}, +\overrightarrow{y}, -\overrightarrow{y}\}$. $\overrightarrow{v_1}$ is said to be in the same direction as $\overrightarrow{v_2}$ iff $\overrightarrow{v_1} = \overrightarrow{v_2}$; $\overrightarrow{v_1}$ is said to be in the opposite direction of $\overrightarrow{v_2}$ iff $\overrightarrow{v_1} = -\overrightarrow{v_2}$; $\overrightarrow{v_1}$ is said to be perpendicular to $\overrightarrow{v_2}$ iff $\overrightarrow{v_1} \cdot \overrightarrow{v_2} = 0$.

***Definition* 4 (State of Containers).** The *state* of a container is given by $((x, y), t, \overrightarrow{v})$, where (x, y) represents the location address on the mesh layout, and t represents the discrete time points of the container, and $\overrightarrow{v} \in \{+\overrightarrow{x}, -\overrightarrow{x}, +\overrightarrow{y}, -\overrightarrow{y}\}$.

***Definition* 5 (Collision).** Let $((x_1, y_1), t_1, \overrightarrow{v_1})$ denote the state of Container1, and $((x_2, y_2), t_2, \overrightarrow{v_2})$ the status of Container2. Container1 and Container2 are said to have a collision at (x_1, y_1) (or (x_2, y_2)) when $t = t_1$ on the mesh layout if and only if $t_1 = t_2$, $(x_1, y_1) = (x_2, y_2)$ and $\overrightarrow{v_1} \in \{+\overrightarrow{x}, -\overrightarrow{x}, +\overrightarrow{y}, -\overrightarrow{y}\} - \{-\overrightarrow{v_2}\}$.

3 Routing Algorithm without Container Conflicts

Based on the model given in the preceding section, the routing algorithm for the mesh path layout is presented as follows. Let all containers set out from their pick up stations at the same time.

Mesh Routing Algorithm

Case a For a container in the job set M_x. In this case, let the container travel along the row PY_i from (PX_i, PY_i) to (DX_i, DY_i).

Case b For a container in the job set M_y. In this case, let the container travel along the column PX_i from (PX_i, PY_i) to (DX_i, DY_i).

Case c For a container in the job set M_{xy}. In this case, let the container firstly travel along the row PY_i from (PX_i, PY_i) to (DX_i, PY_i). Then let it travel along the column DX_i from (DX_i, PY_i) to (DX_i, DY_i).

The routing algorithm looks simple, and if we let containers travel in this rule at an arbitrary speed, it is very likely to have collisions on the mesh layout. However, as we will show shortly, if we control the time when each container reaches every junction node by regulating the containers' speed, the containers can be delivered on the mesh layout without conflicts.

We let ΔT_{+x} denote the time required for a container to travel through one edge of the mesh along the $+\overrightarrow{x}$ direction. Let ΔT_{-x} (ΔT_{+y}, ΔT_{-y}) be defined similarly. We assume that containers travel at the speed v_{+x}, v_{-x}, v_{+y}, v_{-y} in these four cases respectively.

According to the preceding algorithm, we have the following conclusions.

***Lemma* 1** According to our routing algorithm, there is no conflict between any two containers belonging to the same set A_x (or A_y).

Proof: According to the definition of collision, and the assumption that each container has a distinct origin, it should be clear that there is no conflict in the containers belonging to A_x (or A_y). ∎

***Theorem* 1** Based on the routing algorithm, any container will not run into conflict with other containers on the mesh layout, if the following relations are satisfied.

(a)

$$\frac{lcm(\Delta T_1, \Delta T_2)}{max(\Delta T_1, \Delta T_2)} \geq N, \tag{1}$$

where ΔT_1 and ΔT_2 are any combination from $\{\Delta T_{+x}, \Delta T_{-x}, \Delta T_{+y}, \Delta T_{-y}\}$.

(b)

$$\begin{cases} gcd(\Delta T_{+y}, \Delta T_{+x}) \nmid \Delta T_{-y}, \\ gcd(\Delta T_{+y}, \Delta T_{-x}) \nmid \Delta T_{-y}, \\ gcd(\Delta T_{-y}, \Delta T_{+x}) \nmid \Delta T_{+y}, \\ gcd(\Delta T_{-y}, \Delta T_{-x}) \nmid \Delta T_{+y}, \\ gcd(\Delta T_{\pm y}, \Delta T_{\pm x}) \geq N, \end{cases} \tag{2}$$

here *gcd* is the Greatest Common Divisor, and *lcm* is the Least Common Multiple. (cf. Koshy 2002)

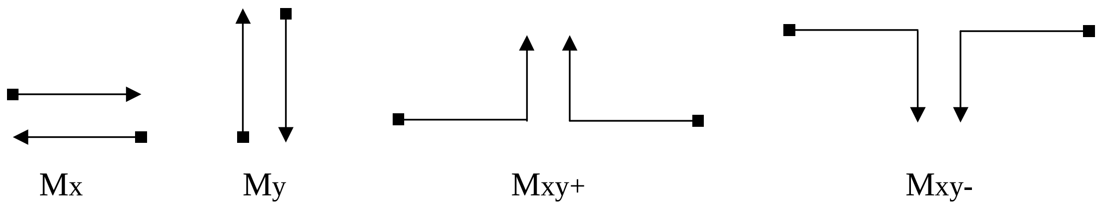

Figure 6: All possible tracks travelled by containers on the mesh path layout.

Proof: According to our mesh routing algorithm, all possible tracks travelled by containers are easily obtained, as shown in figure 6.

Combining all these possible tracks, we obtain all cases of possible conflicts, as shown in figure 7. We omitted a few similar cases, which are symmetrical to some cases shown above.

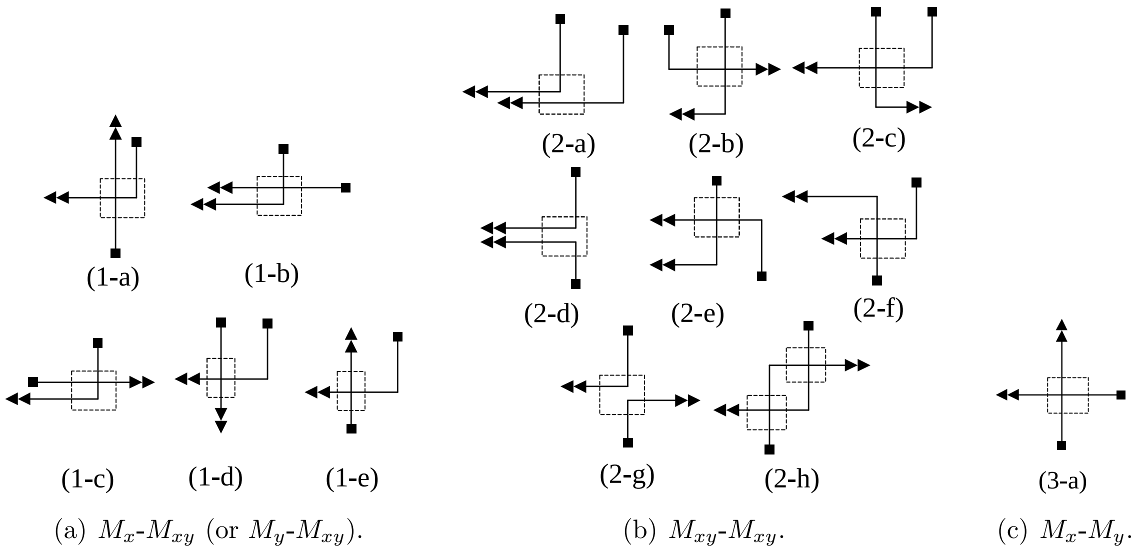

Figure 7: All possible cases of conflicts in routing on the mesh path layout (we omit some other cases that are symmetrical to cases given here).

Assume the initial states of Container1 and Container2 respectively as follows.

Container1: $((x_1, y_1), t_1 = 0, \overrightarrow{v_1})$;

Container2: $((x_2, y_2), t_2 = 0, \overrightarrow{v_2})$.

When $(x_1, y_1) = (x_2, y_2) = (x', y')$, the states of Container1 and Container2 are respectively, Container1: $((x', y'), t_1', \overrightarrow{v_1}')$;

Container2: $((x', y'), t_2', \overrightarrow{v_2}')$.

Now let us prove that $t_1' \neq t_2'$ in all cases of potential conflicts.

Case (1) This case covers (1-a), (1-b), (1-c), (2-d), (2-g), and (3-a) in figure 7. We have the following relations.

$$\begin{cases} t_1' = t_1 + i\Delta T_1 = i\Delta T_1, \\ \\ t_2' = t_2 + j\Delta T_2 = j\Delta T_2, \end{cases}$$

where $0 \leq i, j \leq N-1$, and ΔT_1 and ΔT_2 are any combination from $\{\Delta T_{+x}$, ΔT_{-x}, ΔT_{+y}, $\Delta T_{-y}\}$.

According to the definition of lcm, we have

$$\begin{cases} i_{min} = \frac{lcm(\Delta T_1, \Delta T_2)}{\Delta T_1}, \\ \\ j_{min} = \frac{lcm(\Delta T_1, \Delta T_2)}{\Delta T_2}. \end{cases}$$

According to inequality (1), we know $i_{min}, j_{min} \geq N$, which contradicts with the condition that $0 \leq i, j \leq N-1$. Therefore in all these cases, $t_1' \neq t_2'$ for any i and j, where $0 \leq i, j \leq N-1$.

Case (2) This case covers (1-e), (2-a), (2-b), (2-c), (2-e), (2-f), and (2-h) in figure 7. We have the following relations.

$$\begin{cases} t_1' = i\Delta T_{+y} + j\Delta T_{+x}, \\ \\ t_2' = k\Delta T_{-y}, \end{cases}$$

or

$$\begin{cases} t_1' = i\Delta T_{+y} + j\Delta T_{-x}, \\ \\ t_2' = k\Delta T_{-y}, \end{cases}$$

or

$$\begin{cases} t_1' = i\Delta T_{-y} + j\Delta T_{+x}, \\ \\ t_2' = k\Delta T_{+y}, \end{cases}$$

or

$$\begin{cases} t'_1 = i\Delta T_{-y} + j\Delta T_{-x}, \\ \\ t'_2 = k\Delta T_{+y}, \end{cases}$$

where $0 \leq i, j, k \leq N-1$.

These four relations are similar to each other, so we will focus on the first one.

Consider the following equation.

$$x\Delta T_{+y} + y\Delta T_{+x} = k\Delta T_{-y}, \tag{3}$$

where x and y are integers.

From relation (2), we have

$$gcd(\Delta T_{+y}, \Delta T_{+x}) \nmid k\Delta T_{-y}.$$

By applying basic number theory (Koshy 2002), we know that equation (3) has no integer solutions, so for any $i, j, k \in [0, N-1]$, $t'_1 \neq t'_2$.

Case (3) This case covers (1-d) in figure 7. We have the following relation.

$$\begin{cases} t'_1 = i\Delta T_{+y} + j\Delta T_{+x}, \\ \\ t'_2 = k\Delta T_{+y}. \end{cases}$$

In order to prove that $t'_1 \neq t'_2$, we need to show that

$$i\Delta T_{+y} + j\Delta T_{+x} \neq k\Delta T_{+y},$$

namely, $j\Delta T_{+x} \neq |k-j|\Delta T_{+y}$.

We know that $0 \leq |k-j| \leq N-1$, then this situation can be reduced to an instance of case (1) that we have already proved.

Therefore, we conclude that in all cases, there is no conflict using our routing algorithm under the specified criteria. ∎

4 Analysis of Routing Efficiency

For the distance travelled by containers in our mesh routing algorithm, the following result is easily obtained according to our routing algorithm.

***Lemma* 2** The distance travelled by all containers in our routing algorithm for the mesh path layout is the ideal shortest distance.

Next, we will analyze the time requirement of our mesh routing algorithm.

***Theorem* 2** The time requirement T_r for all containers to transport all jobs is upper-bounded by

$$2(N-1)max\{\Delta T_{+x}, \Delta T_{-x}, \Delta T_{+y}, \Delta T_{-y}\}.$$

Proof: Since all jobs are carried out in parallel, the time requirement for a job set M is determined by the most time-consuming job in the set. Formally, for any given job set, we have

$T_r = max \; \{T((PX_1,\, PY_1),\, (DX_1,\, DY_1)),\, T((PX_2,\, PY_2),\, (DX_2,\, DY_2)),\, \cdots,\, T((PX_k,$ $PY_k),\, (DX_k,\, DY_k))\}$,

where $T((PX_i, PY_i), (DX_i, DY_i))$ is the time requirement for the ith container to complete its job.

Assume that there exists job $((1,1),(N,N))$ which uses the most time and use $max\{\Delta T_{+x},$ $\Delta T_{-x},\, \Delta T_{+y},\, \Delta T_{-y}\}$ time to go through one edge on the mesh layout, then we obtain the following relation.

$T((1,1),(N,N)) = 2(N-1)\; max\{\Delta T_{+x},\, \Delta T_{-x},\, \Delta T_{+y},\, \Delta T_{-y}\}$.

Thus, the time requirement for a job set is upper-bounded by

$T_r \leq T((1,1),\, (N,N)) = 2(N-1)\; max\{\, \Delta T_{+x},\, \Delta T_{-x},\, \Delta T_{+y},\, \Delta T_{-y}\}$. ■

Although our routing algorithm guarantees collision-freedom under some special criteria, the control system needs to know the time point when each container goes through every junction node. So it is necessary for us to consider the relation between different time points when each container goes through every junction node.

***Definition* 6 (Time Difference).** The time difference is the difference of two time points when two containers reach a given junction node. The minimum time difference is the minimum time difference for all containers at every junction node on the mesh layout.

The definition of minimum time difference is quite important for routing control, since it is related to some realistic considerations, such as the length of container carrier, time required to make turns, and crane operation time, etc.

***Theorem* 3** The minimum time difference on the mesh layout is lower-bounded by
$min\{\ gcd(\ \Delta T_{+x},\ \Delta T_{-x}),\ gcd(\ \Delta T_{+x},\ \Delta T_{+y}),\ gcd(\ \Delta T_{+x},\ \Delta T_{-y}),\ gcd(\ \Delta T_{-x},\ \Delta T_{+y}),\ gcd(\ \Delta T_{-x},\ \Delta T_{-y}),\ gcd(\Delta T_{+y},\ \Delta T_{-y})\}$,
namely,
$min_{w,z\in S,w\neq z}\{gcd(w,z)\}$, where $S = \{\ \Delta T_{+x},\ \Delta T_{+y},\ \Delta T_{-x},\ \Delta T_{-y}\}$.

Proof: To get the minimum time difference, we can find the minimum of the following value:

$$i\Delta T_1 + j\Delta T_2,$$

where i, j are integers, and ΔT_1, ΔT_2 are any two numbers from $\{\Delta T_{+x}, \Delta T_{+y}, \Delta T_{-x}, \Delta T_{-y}\}$.

By basic number theory (Koshy 2002), the least positive integer of the form $i\Delta T_1 + j\Delta T_2$ is $gcd(\Delta T_1, \Delta T_2)$.

Thus for any ΔT_1 and ΔT_2 from $\{\Delta T_{+x},\ \Delta T_{+y},\ \Delta T_{-x},\ \Delta T_{-y}\}$, the minimum of the time difference is

$min\{gcd(\ \Delta T_{+x},\ \Delta T_{-x}),\ gcd(\ \Delta T_{+x},\ \Delta T_{+y}),\ gcd(\ \Delta T_{+x},\ \Delta T_{-y}),\ gcd(\ \Delta T_{-x},\ \Delta T_{+y}),\ gcd(\ \Delta T_{-x},\ \Delta T_{-y}),\ gcd(\ \Delta T_{+y},\ \Delta T_{-y})\}$,

namely,

$min_{w,z\in S,w\neq z}\{gcd(w,z)\}$,

where $S = \{\Delta T_{+x}, \Delta T_{+y}, \Delta T_{-x}, \Delta T_{-y}\}$. ∎

5 A Method to Construct ΔT_{+x}, ΔT_{+y}, ΔT_{-x}, ΔT_{-y}

We introduce the following method to construct ΔT_{+x}, ΔT_{+y}, ΔT_{-x}, ΔT_{-y}, which satisfy the criteria of conflict-free routing.

We let $\Delta T_{+y} = P_1^{a+b} P_5^g$, $\Delta T_{-y} = P_2^{c+d} P_5^g$, $\Delta T_{-x} = P_1^a P_2^c P_3^e$, and $\Delta T_{+x} = P_1^a P_2^c P_4^f$, where P_1,P_2,P_3, P_4, and P_5 are primes, and satisfy the following relations: $a, b \geq log_{P_1} N$; $c, d \geq log_{P_2} N$; $e \geq log_{P_3} N$; $f \geq log_{P_4} N$; $g \geq log_{P_5} N$.

***Theorem* 4** The values of ΔT_{+x}, ΔT_{+y}, ΔT_{-x}, ΔT_{-y} constructed by this method satisfy the criteria of conflict-free routing.

Proof: By basic number theory (Koshy 2002), we have

$$\begin{aligned}\frac{lcm(\Delta T_{+y}, \Delta T_{-y})}{\Delta T_{+y}} &= \frac{\Delta T_{+y} \cdot \Delta T_{-y}}{gcd(\Delta T_{+y}, \Delta T_{-y}) \cdot \Delta T_{+y}} \\ &= \frac{\Delta T_{-y}}{gcd(\Delta T_{+y}, \Delta T_{-y})} \\ &= \frac{P_2^{c+d} P_5^g}{P_5^g} = P_2^{c+d} \geq N^2 \geq N.\end{aligned}$$

Similarly we can prove that for any ΔT_1 and ΔT_2 from $\{\Delta T_{+x}, \Delta T_{+y}, \Delta T_{-x}, \Delta T_{-y}\}$, the following relation is satisfied.

$$\frac{lcm(\Delta T_1, \Delta T_2)}{\Delta T_1} \geq N.$$

According to the construction method, we have

$$gcd(\Delta T_{+y}, \Delta T_{+x}) = gcd(P_1^{a+b} P_5^g, P_1^a P_2^c P_4^f)$$

$$= P_1^a \geq N.$$

Because $P_1^a \nmid P_2^{c+d} P_5^g$, we have

$$gcd(\Delta T_{+y}, \Delta T_{+x}) \nmid \Delta T_{-y}.$$

Similarly we can prove that the relation (2) is satisfied.

Therefore the values of ΔT_{+x}, ΔT_{+y}, ΔT_{-x}, ΔT_{-y} constructed by this method satisfy all the criteria of conflict-free routing. ■

The differences in ΔT_{+x}, ΔT_{+y}, ΔT_{-x}, ΔT_{-y} have implications on the routing control. The smaller value of this difference generally means the more accurate timing when containers arrive at the junctions.

***Proposition* 1** The minimum time difference of ΔT_{+x}, ΔT_{+y}, ΔT_{-x}, ΔT_{-y} constructed by the above method is $min\{P_1^a, P_2^c, P_5^g\}$, which is lower-bounded by $\Omega(N)$.

Proof: According to Theorem 3, the minimum time difference on the mesh layout is lower-bounded by

$min\{\ gcd(\ \Delta T_{+x},\ \Delta T_{-x}),\ gcd(\ \Delta T_{+x},\ \Delta T_{+y}),\ gcd(\ \Delta T_{+x},\ \Delta T_{-y}),\ gcd(\ \Delta T_{-x},\ \Delta T_{+y}),$ $gcd(\ \Delta T_{-x},\ \Delta T_{-y}),\ gcd(\ \Delta T_{+y},\ \Delta T_{-y})\}$.

Substituting into the values of ΔT_{+x}, ΔT_{+y}, ΔT_{-x}, ΔT_{-y}, we have

$min\{\ gcd(\ \Delta T_{+x},\ \Delta T_{-x}),\ gcd(\ \Delta T_{+x},\ \Delta T_{+y}),\ gcd(\ \Delta T_{+x},\ \Delta T_{-y}),\ gcd(\ \Delta T_{-x},\ \Delta T_{+y}),$ $gcd(\ \Delta T_{-x},\ \Delta T_{-y}),\ gcd(\ \Delta T_{+y},\ \Delta T_{-y})\}$

$= min\{P_1^a \cdot P_2^c, P_1^a, P_2^c, P_1^a, P_2^c, P_5^g\}$

$= min\{P_1^a, P_2^c, P_5^g\}$.

Therefore, the minimum time difference of ΔT_{+x}, ΔT_{+y}, ΔT_{-x}, ΔT_{-y} constructed by this method is lower-bounded by $min\{P_1^a, P_2^c, P_5^g\}$. Because $P_1^a, P_2^c, P_5^g \geq N$, the minimum time difference is lower-bounded by $\Omega(N)$. ■

From Theorem 2, the largest value of ΔT_{+x}, ΔT_{+y}, ΔT_{-x}, ΔT_{-y} generally means a higher task completion time (for the given choice of time unit). The following result bounds this value.

***Theorem* 5** The values of ΔT_{+x}, ΔT_{+y}, ΔT_{-x}, ΔT_{-y} are bounded by $\Theta(N^3)$.

Proof: To keep the values of ΔT_{+x}, ΔT_{+y}, ΔT_{-x}, ΔT_{-y} as small as possible, we let $a = b = \lceil log_{P_1} N \rceil$; $c = d = \lceil log_{P_2} N \rceil$; $e = \lceil log_{P_3} N \rceil$; $f = \lceil log_{P_4} N \rceil$; $g = \lceil log_{P_5} N \rceil$.

For ΔT_{+y}, we have $log_{P_1} N \leq a, b \leq log_{P_1} N + 1$ and $log_{P_5} N \leq g \leq log_{P_5} N + 1$. So we have the following relations: $N \leq P_1^a, P_1^b \leq P_1 \cdot N$ and $N \leq P_5^g \leq P_5 \cdot N$. Therefore, we have $N^3 \leq P_1^{a+b} P_5^g \leq P_1^2 P_5 \cdot N^3$, namely $N^3 \leq \Delta T_{+y} \leq P_1^2 P_5 \cdot N^3$. Since $P_1^2 P_5$ is a constant, we obtain that $\Delta T_{+y} = \Theta(N^3)$.

Similarly, we can prove that $\Delta T_{+x} = \Delta T_{-x} = \Delta T_{-y} = \Theta(N^3)$. ■

We give a simple example to illustrate the construction. Let $N = 7$, we can choose $\Delta T_{+y} = 2^{3+3} \cdot 13$, $\Delta T_{-y} = 3^{2+2} \cdot 13$, $\Delta T_{-x} = 2^3 \cdot 3^2 \cdot 7$, $\Delta T_{+x} = 2^3 \cdot 3^2 \cdot 11$. The minimum time difference of this case is $2^3 = 8$, and the ratio between the maximum and the minimum of $\Delta T_{+x}, \Delta T_{+y}, \Delta T_{-x}, \Delta T_{-y}$ is about 2.

6 Numerical Experiments

Numerical experiments have been conducted on computer based on our model of container routing in the mesh yard. In each run, the IDs of the pickup and drop-off stations are randomly generated, and each job has a distinct origin and also a distinct (but different) destination. In the simulations, we observe the minimum time differences for yards with different sizes and different numbers of jobs (or containers). Meanwhile, we also compare the time requirement of our scheme with that of two other schemes.

6.1 Observations of Minimum Time Differences

The notation of minimum time difference is important for routing control, since it is related to some realistic considerations, such as the length of container carriers, time required to make turns, and crane operation time. We will investigate the impact of the size of the yard layout and the number of jobs (or containers) on minimum time difference.

We choose $P_1 = 2$, $P_2 = 3$, $P_3 = 5$, $P_4 = 7$, and $P_5 = 11$. According to our construction in Section 5, the values of ΔT_{+y}, ΔT_{-y}, ΔT_{+x} and ΔT_{-x} are calculated as shown in table 1 for different yard sizes: $N = 7$, $N = 9$, $N = 11$, and $N = 13$.

	ΔT_{+y}	ΔT_{-y}	ΔT_{+x}	ΔT_{-x}
N=7	$2^{3+3} \times 11 = 704$	$3^{2+2} \times 11 = 891$	$2^3 \times 3^2 \times 7 = 504$	$2^3 \times 3^2 \times 5^2 = 1800$
N=9	$2^{4+4} \times 11 = 2816$	$3^{2+2} \times 11 = 891$	$2^4 \times 3^2 \times 7^2 = 7056$	$2^4 \times 3^2 \times 5^2 = 3600$
N=11	$2^{4+4} \times 11 = 2816$	$3^{3+3} \times 11 = 8019$	$2^4 \times 3^3 \times 7^2 = 21168$	$2^4 \times 3^3 \times 5^2 = 10800$
N=13	$2^{4+4} \times 11^2 = 30976$	$3^{3+3} \times 11^2 = 88209$	$2^4 \times 3^3 \times 7^2 = 21168$	$2^4 \times 3^3 \times 5^2 = 10800$

Table 1: Constructed ΔT_{+y}, ΔT_{-y}, ΔT_{+x} and ΔT_{-x} for different yard sizes: $N = 7$, $N = 9$, $N = 11$, and $N = 13$.

Firstly, we observe the minimum time difference of each junction for the case when yard size $N = 7$ and job number $|M| = 24$. The generation of the job set is shown in table 2, and the corresponding minimum time differences for all junctions are shown in table 3. In table 3, $NULL$ means that there is no container passing that junction. If there is only one container passing a junction, the minimum time difference in table 3 is equal to the time required for this container to reach the junction. The smallest minimum time difference for all junctions happens in the junction $(2, 4)$, and it means that the interval time required for avoiding container conflict at the junction should not exceed 99 time units. If we let a time unit equal to a second, the minimum time difference in the case $N = 7$ is 99 seconds. From Figure 12, the time requirement to finish 24 jobs in a 7×7 mesh layout is 13616 seconds. Since $\frac{3600 \times 24}{13616} \times 24 \approx 152$, around 152 jobs can be finished for a 7×7 mesh layout in one day.

We also run the simulations and get the minimum time difference of all junctions for cases $N = 7$, $N = 9$, $N = 11$, and $N = 13$. We consider different job numbers $|M| = 6$, $|M| = 15$, and $|M| = 24$ for each case. The simulation result is shown in figure 8.

From figure 8, we know that the minimum time difference decreases as the number of jobs increases for a given yard layout. Also, if we fix the number of jobs, the minimum time difference increases as N increases.

6.2 Comparisons of Time Requirement with Other Schemes

In this subsection, we compare our scheme with two other schemes: Greedy and Sorting schemes (Qiu and Hsu 2000). Firstly, we briefly review these two schemes.

Job ID	**Source**	**Destination**
1	(7,2)	(5,5)
2	(5,4)	(6,7)
3	(6,6)	(2,7)
4	(4,1)	(2,1)
5	(6,4)	(6,2)
6	(5,6)	(1,7)
7	(7,5)	(4,3)
8	(2,3)	(4,2)
9	(7,7)	(4,5)
10	(3,6)	(5,1)
11	(1,3)	(7,6)
12	(6,1)	(3,7)
13	(2,2)	(5,2)
14	(2,5)	(1,2)
15	(3,2)	(2,4)
16	(1,5)	(1,1)
17	(4,7)	(7,3)
18	(4,4)	(2,6)
19	(5,3)	(3,1)
20	(3,4)	(7,4)
21	(6,5)	(6,3)
22	(4,6)	(5,7)
23	(3,3)	(7,1)
24	(3,5)	(1,4)

Table 2: Generation of job set for the case when $N = 7$ and $|M| = 24$.

(a) **Greedy scheme:** The greedy routing algorithm always makes the choice that looks best at the moment. The greedy routing algorithm has been widely used in vehicle routing and material handling problem (Broadbent *et al.* 1985, Huang *et al.* 1989, Kim and Tanchoco 1991, Kim and Tanchoco 1993). For Greedy scheme in our experiment, each container is routed to its destination through the shortest path. When multi containers arrive at the same junction within a short time range, they will form a queue before the junction. We assume that there is some centralized mechanism to avoid the deadlock for Greedy scheme. In order to avoid

Row or Column ID of junction	1	2	3	4	5	6	7
1	7200	891	3600	504	1008	1512	4068
2	1782	1800	3600	99	NULL	574	891
3	891	704	4896	792	1800	1021	891
4	187	3208	927	5400	2673	792	891
5	1408	1800	2522	NULL	2088	1008	117
6	5712	3096	792	288	387	2512	808
7	6416	NULL	504	792	1512	1800	NULL

Table 3: Minimum time difference of each junction for the case when $N = 7$ and $|M| = 24$.

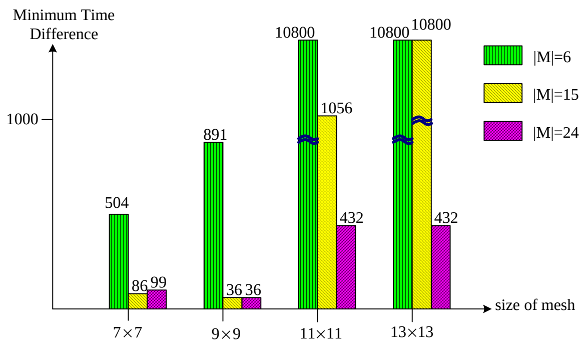

Figure 8: Minimum time difference for different cases.

conflicts with preceding containers and ensure them to leave the junction safely, each container in a queue must wait for at least a fixed interval time after all its preceding containers leave the queue. In our simulation, we let the fixed interval time equal to the minimum time difference.

(b) **Sorting scheme:** The sorting scheme for moving cargo in material handling systems has been proposed by Qiu and Hsu (2000), which adapts the data packet sorting algorithm (Leighton 1992). The sorting scheme for transferring containers in our simulation are as follows:

Sorting Algorithm for Transferring Containers

Step 1 Move the containers in each column and make sure that at most one container in each row is destined for each column.

Step 2 Move each container along its row to its correct column.

Step 3 Move each container along its column to its correct row, which is also its final destination.

We run the simulations and compare our scheme with Greedy and Sorting schemes. We consider the impact of different N and job number on the time requirement to deliver all containers. We let the container speed in Greedy and Sorting schemes equal to the average speed of $+y$, $-y$, $+x$, and $-x$ directions in our scheme.

Firstly, the time requirements of all three schemes for different job number when $N = 9$ are examined, as shown in figure 9. Based on the comparisons, we can see that the greedy scheme has the minimum required time, and the sorting scheme has the most required time. The required time of our scheme is only a bit larger than that of the greedy scheme, compared with that of the sorting scheme. However, the greedy scheme offers no conflict-free guarantee, and it could even cause deadlock (cf. figure 10).

Figure 11 shows that containers form queues with the size of [0,5] at junctions in Greedy scheme. On the contrary, the queue size of Sorting and our schemes is 0. Therefore, our scheme requires less control overhead compared with greedy scheme, since the greedy scheme require a mechanism to avoid deadlock and manage the container queue. Another advantage that allows our scheme to outperform the sorting scheme is that containers in our schemes travel the shortest path, while extra distance is needed in the sorting scheme.

Secondly, we compare the required time of the three schemes for different N under the same job number $|M| = 24$, as shown in figure 12. We can find that as N increases, the time requirement for all three schemes increases. In each run, the time requirement of our scheme is closer to the greedy scheme than that of the sorting scheme.

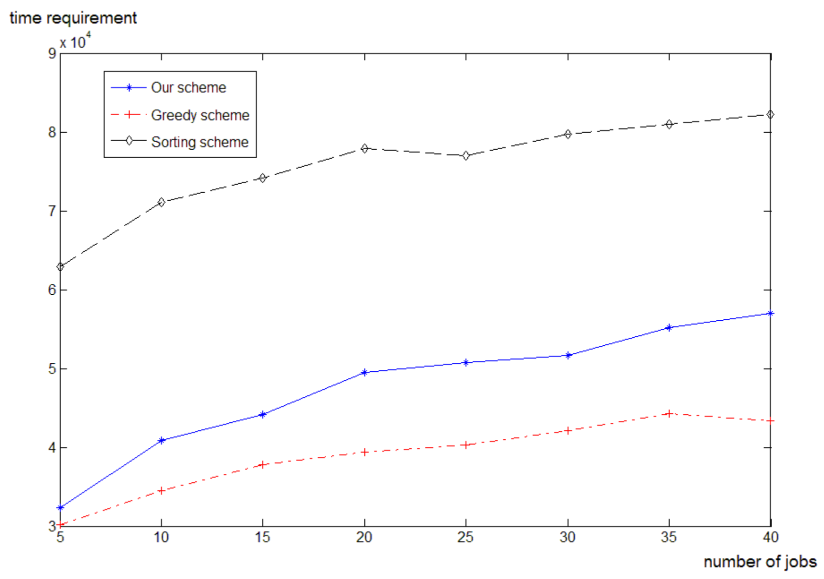

Figure 9: Comparisons of time requirements in three schemes when $N = 9$.

7 Discussions and Conclusions

We have presented a mathematical model for container routing in the mesh yard layout. Based on this model and the discrete time division, we proposed a routing algorithm which allows containers to travel at different multiples of the unit time along different directions, which guarantees the freedom of conflicts. The timing control requirement was analyzed, and the method to construct the multiples of the unit time was also introduced. Numerical results show that our scheme has good performance with the freedom of conflicts. Our contribution lies in the model and method to generate a conflict-free schedule for container routing.

With our mesh routing algorithm, all the containers can move directly towards their destinations without conflicts. Therefore, the overall routing performance is ensured. Moreover, since each container makes at most one turn during the entire routing process, the speed of each containers is changed no more than once during the process. Therefore, the energy requirement by our routing algorithm is also relatively low. In our routing model, each container on the mesh topology is assumed to be one point. However, there are some details that we must consider in actual implementations, such as the size of

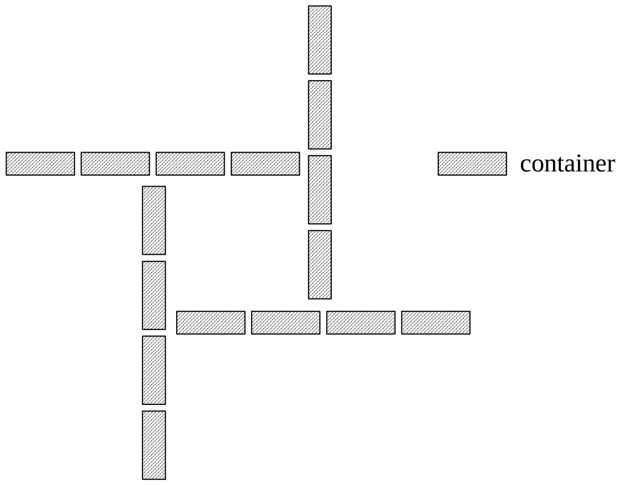

Figure 10: Possible deadlock in the greedy scheme.

junction, the length of the container, etc. These considerations impose a minimum time difference, which can be adjusted by the control system. According to Proposition 1, the minimum time difference is decided by $min\{P_1^a, P_2^c, P_5^g\}$. Thus we can choose the value of $\{P_1^a, P_2^c, P_5^g\}$ to increase the minimum time difference. Therefore, the abstract model and the routing algorithm can be applied to actual mesh-like layout. Similarly, the task completion time can be controlled by choice of the units of time, the distance between intersections and/or the speeds of containers. According to our method for constructing $\Delta T_{+x}, \Delta T_{+y}, \Delta T_{-x}, \Delta T_{-y}$ and Theorem 2, as N increases, the time requirement to finish the jobs seems to increase quickly. However, we can choose a small unit of time to keep the actual time requirement low, as long as the minimum time difference for avoiding conflicts is satisfied.

We assumed that when a container reaches its destination, it enters the buffer and leaves the mesh grid. This assumption can also be relaxed. Usually, when a container enters the buffer of the P/D station, it takes some time for the container to completely leave the mesh grid. The situation is similar when a container goes through the junction. However, as long as the time required for a container to enter the buffer of the station is less than the minimum time difference, there are still no conflicts during the container routing.

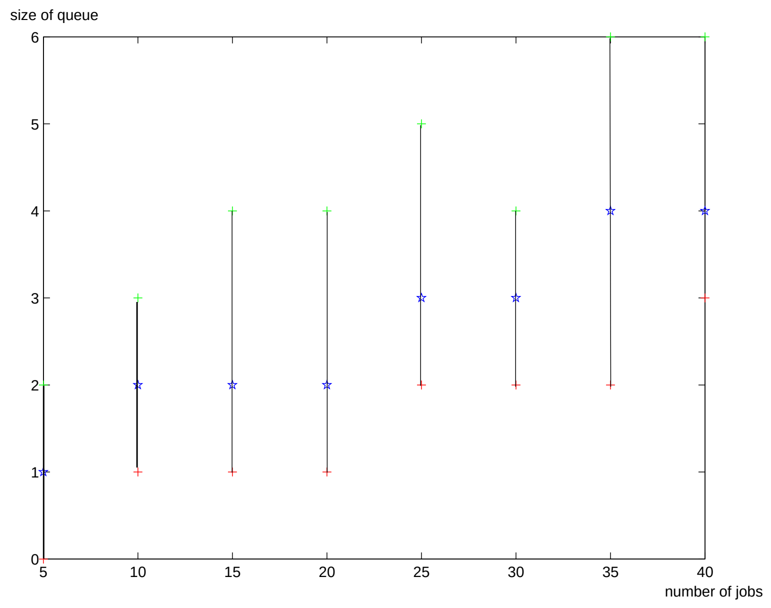

Figure 11: The queue size of Greedy scheme when $N = 9$.

In our routing scheme, we let containers in one direction travel at a constant speed. Although it is possible to vary the container speed according to different congestion situation over the mesh yard, it also requires more complicated control mechanism and more difficult to implement. On the other hand, our routing scheme is easy to implement, require less control overhead, and can still achieve the good performance.

In the abstract container routing model for the mesh yards, we assume that all blocks have equal sizes, and the P/D station is located at same position as the junction. This assumption can be relaxed to other cases for different locations of P/D stations, and for blocks with different sizes, as long as the regularity of the whole yard layout is guaranteed. The reason is that under the regular topology, we can still apply the same number theory method to theoretically analyze the conflict problem of the container routing in the mesh yards.

Future studies could attempt to address a variety of issues: first, in our current scheme, a single blockage will cause the failure of the entire system. It is, therefore, important to consider fault-tolerant strategies. Second, our algorithm can only process batched, cyclic jobs. However, in the realistic terminal operation, the jobs arise continuously. So the

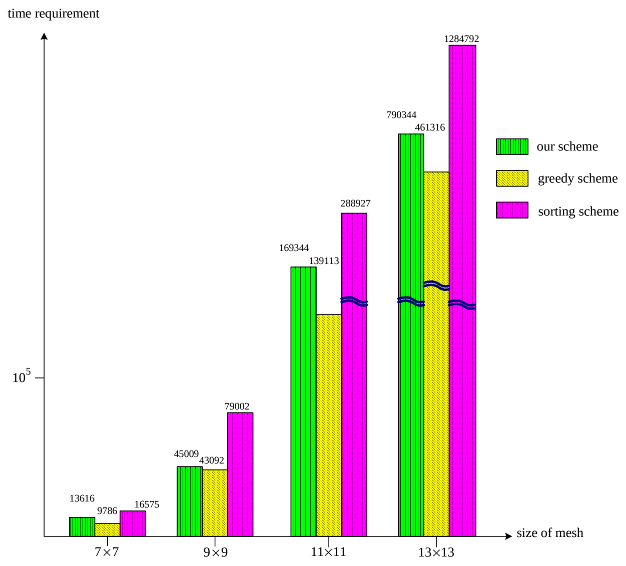

Figure 12: Comparisons of time requirements in three schemes when $|M| = 24$.

extension of our scheme to the dynamic job process has great significance. This extension may be implemented by dividing time into several time windows, and regard routing in each time window as a static routing problem as we consider here. Third, we need to devise a method to decide the number of container carriers for the given jobs and to deal with idle container carriers, such as AGVs, shuttles, or platforms.

Acknowledgment

We would like to thank all referees for their very helpful comments and suggestions. We acknowledge the Maritime and Port Authority, A*STAR and Nanyang Technological University, all of Singapore, for their support of the research project.

References

[BROADBENT *et al.* 1985] BROADBENT, A.-J., BESANT, C.-B., PREMI, S.-K., and WALKER, S.-P., 1985, Free ranging AGV systems: promise, problems and pathways. *Proceeding of the 2nd International Conference on Automated Materials Handling*, 221–237.

[CHAN *et al.* 2000] CHAN, C.-H., 2000, Dyanmic AGV-container job deployment stragegy. Master's thesis, High Performance Computation for Engineered Systems, Sinpore-MIT Alliance.

[CHEN *et al.* 2003] CHEN, C., HUANG, S.-Y., HSU, W.-J., TOH, A.-C., and LOH, C.-K., 2003, Platform-based AS/RS for container storage. *Proceedings of the 2003 IEEE International Conference on Robotics and Automation (ICRA 2003).*

[CHEN 2003] CHEN, C., 2003, Simulation and optimization of container yard operations: a survey. *Proceedings of International Conference on Port and Maritime R and D and Technology*, 23–29.

[HUANG and HSU 1994] HUANG, S.-Y., and HSU, W.-J., 1994, Routing automated guided vehicles on mesh like topologies. *Proceedings of International Conference on Automation, Robotics and Computer Vision*, Paris.

[HUANG *et al.* 1989] HUANG, J., PALEKAR, U.-S., and KAPOOR, S.-G., 1989, A labeling algorithm for the navigation of automated guided vehicles. *Advances in Manufacturing Systems Engineering, Proceedings of the ASME winter annual meeting*, San Francisco, California, USA, 181–193.

[IOANNOU *et al.* 2000] IOANNOU, P.-A. , JULA, H., LIU, C.-I., VUKADINOVI, K., and POURMOHAMMADI, H., 2000, Advanced material handling: Automated guided vehicles in agile ports. Research report, Center for Advanced Transportation Technologies, University of Southern California, Los Angeles.

[KOSHY 2002] KOSHY, T., 2002, *Elementary number theory with application*, Harcourt/Academic Press.

[KIM *et al.* 2000] KIM, K.-H., PARK, Y.-M., and RYU, K.-R., 2000, Deriving decision rules to locate export containers in container yards. *European Journal of Operational Research*, 124:89–101.

[KIM *et al.* 1991] KIM, C.-W, and TANCHOCO, J.-M.-A., 1991, Conflict-free shortest-time bi-directional AGV routing. *International Journal of Production Research*, 29(12):2377–2391.

[KIM and TANCHOCO 1993] KIM, C.-W, and TANCHOCO, J.-M.-A., 1993, Operational control of a bi-directional automated guided vehicle systems. *International Journal of Production Research*, 31(9):2123–2138.

[LEIGHTON 1992] LEIGHTON, F.-T., 1992, *Introduction to parallel algorithms and architectures : arrays, trees, hypercubes*, Morgan Kaufmann Publishers, San Mateo, California.

[LIU *et al.* 2002] LIU, C.-I., JULA, H., and IOANNOU, P.A., 2002, Desigh, simulation, and evaluation of automated container terminals. *IEEE Transactions on Intelligent Transportation Systems*, 3(1):12–26.

[QIU and HSU 2000] QIU, L. and HSU, W.-J., 2000, Routing AGVs on a mesh-like path topology. *Proceedings of the IEEE Intelligent Vehicles Symposium 2000 (IVS 2000)*, Dearborn, Michigan, USA , 392–397.

[QIU and HSU 2001] QIU, L., and HSU, W.-J., 2001, A bi-directional path layout for conflict-free routing of AGVs. *International Journal of Production Research*, 39(10):2177–2195.

[QIU *et al.* 2002] QIU, L., HSU, W.-J., HUANG, S.-Y., and WANG, H., 2002, Scheduling and routing algorithms for AGVs: a survey. *International Journal of Production Research*, 40(3):745–760.

[ZENG and HSU 2003] ZENG, J., and HSU, W.-J., 2003, Conflict-free routing of AGVs on the mesh topology based on a discrete-time model. *Proceedings of the 2003 IEEE International Conference on Robotics and Automation (ICRA 2003)*. Taipei, Taiwan.